MASTER COPY KEEP THIS COPY FOR REPRODUCTION PURPOSES

AD-A276 255

CUMENTATION PAGE

Form Approved
OMB No 0704-0188

...ation is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources ...pleting and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this ... reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson ...2, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

2. REPORT DATE
November 15, 1993

3. REPORT TYPE AND DATES COVERED
Final 27 Aug 90 - 26 Aug 93

4. TITLE AND SUBTITLE
Design and Analysis of Scalable Parallel Algorithms

5. FUNDING NUMBERS
DAAL03-90-G-0201

6. AUTHOR(S)

Vipin Kumar

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Computer Science Department
University of Minnesota
Minneapolis, MN 55455

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
ARO 28408.1-MA-SDI

11. SUPPLEMENTARY NOTES
The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The objective of this research is to develop efficient parallel algorithms for a variety of problems and to analyze the scalability of new and existing parallel algorithms. Scalability analysis is an important tool used for predicting the performance of an algorithm-architecture combination when one or more of the hardware related parameters (interconnection network, speed of processors, speed of communication channels, number of processors) are changed. The problems studied as a part of this project come from diverse domains such as solution of differential equations, discrete optimization, neural network based learning, sorting and graph algorithms. In particular, we have studied parallel algorithms for solving linear systems using the preconditioned conjugate gradient method, partitioning of finite element meshes, balancing load in unstructured tree search arising in discrete optimization, the backpropagation neural network learning algorithm, dynamic programming, fast fourier transform, sorting, shortest-path computation for graphs, robot motion planning, and matrix multiplication.

14. SUBJECT TERMS
Parallel Algorithms, Scalability Analysis, Isoefficiency

15. NUMBER OF PAGES
12

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500 Standard Form 298 (Rev 2-89)
Prescribed by ANSI Std Z39-18
298-102

94 2 24 130

94-06167

DESIGN AND ANALYSIS OF SCALABLE PARALLEL ALGORITHMS

Final Report

Vipin Kumar

November 15, 1993
U. S. Army Research Office

Contract / Grant Number: 28408-MA-SDI

Computer Science Department
University of Minnesota
Minneapolis, MN 55455

Approved for Public Release;
Distribution Unlimited.

Contents

Abstract

The objective of this research is to develop efficient parallel algorithms for a variety of problems and to analyze the scalability of new and existing parallel algorithms. Scalability analysis is an important tool used for predicting the performance of an algorithm-architecture combination when one or more of the hardware related parameters (interconnection network, speed of processors, speed of communication channels, number of processors) are changed. The problems studied as a part of this project come from diverse domains such as solution of differential equations, discrete optimization, neural network based learning, sorting and graph algorithms. In particular, we have studied parallel algorithms for solving linear systems using the preconditioned conjugate gradient method, partitioning of finite element meshes, balancing load in unstructured tree search arising in discrete optimization, the backpropagation neural network learning algorithm, dynamic programming, fast fourier transform, sorting, shortest-path computation for graphs, robot motion planning, and matrix multiplication.

1 Problem Statement

At the current stage of technology, it is possible to construct cheap but extremely powerful parallel computers by simply interconnecting a large number of sequential computers. But utilizing the massive power of these systems has been very difficult. One reason for this difficulty is the lack of availability of scalable parallel algorithms for a wide variety of problems.

This project resulted in development of new and more comprehensive analytical tools to study the scalability of parallel algorithms and architectures. New metrics were developed to characterize the scalability of algorithm-architecture combinations for which currently available metrics are not adequate. Frameworks for studying the impact of technology dependent parameters such as the processor and communication speeds were also studied. Metrics were developed that help in characterizing the cost-effectiveness of parallel architectures in the context of parallel algorithms to be implemented on them. The project also focused on developing parallel algorithms and data structures for a variety of numeric and nonnumeric problems and the analysis of their performance and scalability on various parallel architectures. This sheds light on what problems can be solved cost effectively on large-scale parallel computers. This also gives us insights into the best possible parallel architectures for solving various problems of practical interest. Some of the problems/algorithms investigated in this research are load balancing issues in tree search algorithms for large-scale integer-linear programming problems, matrix algorithms, graph problems, and partial differential equations.

2 Summary of Important Results

Our approach to the design of scalable parallel algorithms is unique in that it simultaneously addresses many critical issues in parallel computing. We design parallel algorithms using a small class of basic parallel operations as building blocks. This allows us to design algorithms for parallel architectures having a variety of interconnection topologies and performance characteristics. It also facilitates porting an algorithm between different parallel architectures. We make use of precise scalability and performance metrics. This allows us to perform theoretical analysis based on a realistic characterization of both problem size and hardware characteristics. While such a comprehensive approach to parallel algorithm design is risky because of its complexity, we believe that our methods will enable a rapid growth in the use of parallel computers for computationally intensive applications.

We have developed a scalability metric, called the *isoefficiency function*, which relates the size of the problem to be solved to the number of processors for an increase in speedup in proportion to the number of processors used [6, 17, 20, 21]. The isoefficiency function of an algorithm architecture pair is defined as the rate at which the problem size (W) needs to grow with the number of processors (p) in order to maintain the efficiency at some constant value. We have used the scalability analysis in designing and determining best parallel algorithms and architectures for solving problems such as FFT [11], shortest path [23], matrix multiplication [9], sorting [26] and load balancing schemes used in parallel state space search [15, 22].

In [26], we analyze the scalability of a variety of parallel sorting algorithms on the mesh multicomputers. Many sorting algorithms are originally designed to work for the case in which there is one processor for each element. If there are fewer processors than the number of elements tc be sorted, then the algorithm has to be adapted to work with more than one elements per processor. We show that algorithms derived by simply allowing a processor to multiplex as more than one processor (a technique advocated in the data-parallel programming paradigm [13] leads to poorly scalable parallel algorithms.

In [9], we use the isoefficiency metric to analyze the scalability of some parallel matrix multiplication algorithms on different classes of parallel architectures. We show that the isoefficiency function for a hypercube algorithm is $O(p \log p)$ which is as good as that of the best known algorithm for CRCW-PRAM. The isoefficiency of the best known algorithm on a mesh multicomputer is $O(p\sqrt{p})$. We analyze the performance of three different parallel formulations of matrix multiplication for different values of p and W and predict the conditions under which each formulation is better than the others. We discuss the dependence of scalability on technology dependent factors such as communication and computation speeds and show that under certain conditions, it may be better to have a parallel computer with k-fold as many processors rather than one with the same number of processors, each k-fold as fast.

In [19], we critically assess the state of the art in the theory of scalability analysis, and motivate further research on the development of new and more comprehensive analytical tools to study the scalability of parallel algorithms and architectures. We survey a number of techniques and formalisms that have been developed for studying scalability issues, and discuss their interrelationships. For instance, we show some interesting relationships between the technique of isoefficiency analysis developed in [21] and many other methods for scalability analysis. We point out some of the weaknesses of the existing schemes, and discuss possible ways of extending them.

In [11], we analyze the scalability of the parallel Fast Fourier Transform algorithm on mesh and hypercube connected multicomputers. The scalability analysis of FFT provides several important insights. On the hypercube architecture, parallel FFT algorithm can obtain linearly increasing speedup with respect to the number of processors with orly a moderate increase in problem size. But there is a limit on the achievable efficiency and this limit is determined by the ratio of CPU speed and communication bandwidth of the hypercube channels. Efficiencies higher than this limit can be obtained only if the problem size is increased very rapidly. It is also shown that pipelining the communication, thereby overlapping it with the computation, does not improve the scalability significantly. The scalability analysis for the mesh connected mul.icomputers re veals that FFT cannot make efficient use of large-scale mesh architectures unless the bandwidth of the communication channels is increased as a function of the number of processors. It is shown that addition of features such as cut-through-routing (also known as worm-hole routing) to the mesh architecture do not improve the overall scalability characteristics of the FFT algorithm on this architecture. We also show that under certain assumptions, it is more cost-effective to implement the FFT algorithm on a hypercube rather than a mesh despite the fact that large scale meshes are cheaper to construct than large hypercubes. All the results in [11] hold for ordered and multidimensional FFT as well.

In [10], we analytically determine the optimal number of processors to employ when the criterion of optimality is minimizing the parallel execution time of the algorithm. Each parallel algorithm-architecture combination has a unique overhead function, the value of which depends on the size of the problem being attempted, and the number of processors being employed. We study the impact of this characteristic of a parallel system, the overhead function, on the optimal number of processors to be used for a given problem. We study the behavior of the efficiency obtained while operating at the point of peak performance w.r.t parallel speedup. Flatt and Kennedy [5, 4] did a similar study for certain kinds of overhead functions. Our analysis is an extension, and in some ways a generalization of their analysis. In particular, in this paper we overcome some of the limitations of their analysis. We show how our results relate to those of Flatt and Kennedy [5] and other researchers who have done similar performance analysis of parallel systems [28, 24, 21]. We then study a more general criterion of optimality and show how operating at the optimal point is equivalent to operating at a unique value of efficiency which is characteristic of the criterion of optimality and the properties of the parallel system under study.

In [8, 18, 21, 25], we analyze the scalability of a number of load balancing algorithms which can be applied to problems that have the following characteristics : the work done by a processor can be partitioned into independent work pieces; the work pieces are of highly variable sizes; and it is not possible (or very difficult) to estimate the size of total work at a given processor. For such problems, any load balancing scheme has to distribute work dynamically among different processors. We have been able to determine the most scalable load balancing schemes for different architectures such as hypercube, mesh and network of workstations. For each architecture, we establish lower bounds on the scalability of any possible load balancing scheme. We present the scalability analysis of a number of source and server initiated load balancing schemes. From this we gain valuable insights into which schemes can be expected to perform better under what problem and architecture characteristics. For each of these architectures, we are able to determine near optimal load balancing schemes. In particular, some of the algorithms presented and analyzed here for hypercubes are more scalable than algorithms known earlier. Results obtained from implementation of these schemes in the context of various problems such as optimizing floorplan of a VLSI chip [1], generating test patterns for combinatorial circuits [3] and tautology verification [18, 8, 2] on various machines including a 1024 processor nCUBE2, 128 processor Intel Hypercube, Symult 2010 and BBN Butterfly are used to validate our theoretical results. We have also demonstrated the accuracy and viability of our framework for scalability analysis.

In [7], we investigate the suitability of three techniques for partitioning finite element meshes. These are striped partitioning, scattered decomposition and binary decomposition. We have shown that for a hypercube connected network, striping has an isoefficiency of $O(P^2)$, scattered decomposition has an isoefficiency of $O(P\log^3 P)$ and binary decomposition has an isoefficiency which lies between $O(P\log^2 P)$ and $O(P^2)$ depending on the shapes of the partitions. Analysis for mesh connected architectures can be performed in a similar fashion as presented in this paper. Experimental results presented show that using very simple techniques for optimizing communication volume, binary decomposition can achieve close to best case isoefficiencies. These results indicate that striping is not efficient for large number of processors. The study shows that both binary de-

composition and scattered decomposition techniques perform significantly better for lower values of startup times, whereas striping gains very little in terms of performance by reduced startup times. The performance of striped partitioning improves more significantly than others with decrease in the per-word transfer time, as the volume of communication is much higher in this scheme.

In [12], the performance and scalability of the Preconditioned Conjugate Gradient Algorithm on parallel architectures such as mesh, hypercube and CM5 is analyzed. It is shown that for penta-diagonal matrices resulting from two dimensional finite difference grids, the computation of vector inner products dominates the rest of the computation in terms of communication overheads. However, with a suitable mapping, the parallel formulation of a PCG iteration is highly scalable for such matrices on a machine like the CM-5 whose fast control network practically eliminates the overheads due to inner product computation. The use of the Incomplete Cholesky (IC) preconditioner can lead to further improvement in scalability on the CM-5 by a constant factor. As a result, a parallel formulation of the PCG algorithm with IC preconditioner may execute faster than that with a simple diagonal preconditioner even if the latter runs faster in a serial implementation. For hepta-diagonal matrices resulting from three dimensional finite difference grids, the scalability is quite good on a hypercube or the CM-5, but not as good on a 2-D mesh such as Intel Touchstone machine. In case of a random sparse matrix with a constant number of non-zero elements in each row, the parallel formulation of the PCG iteration is unscalable on any message passing parallel architecture. But the parallel system can be made scalable either if, after re-ordering, the non-zero elements of the $N \times N$ matrix can be confined in a band whose width is $O(N^y)$ for any $y < 1$, or if the number of non-zero elements per row increases as N^x for any $x > 0$. Scalability increases as the number of non-zero elements per row is increased and/or the width of the band containing these elements is reduced. For random sparse matrices, the scalability is asymptotically the same for all architectures. Many of these analytical results are experimentally verified on the CM-5 parallel computer.

In [14], we present new methods for load balancing of unstructured tree computations on large-scale SIMD machines, and analyze the scalability of these and pre-existing schemes. The analysis and experiments show that our new load balancing methods are highly scalable on SIMD architectures. In particular, their scalability is no worse than that of the best load balancing schemes on MIMD architectures. We verify our theoretical results by implementing the 15-puzzle problem on a CM-2[1] SIMD parallel computer.

In [16], we present a new and highly scalable network partitioning method, called *checkerboard*, for mapping the Backpropagation algorithm on a hypercube multicomputer. Our algebraic and experimental analysis shows the superiority of checkerboard over previous network partitioning schemes based on vertical sectioning. We plan to test our parallel formulation on Bignet, which is a neural network learning algorithm, currently being used for the protein folding application[27].

Our research efforts as a part of this project have lead to the publication of a text [17]. This book addresses our algorithm design methodology in detail, and uses many of the above results as practical examples. The text is directed at both students (advanced undergraduates and graduates) and at practicing algorithm designers and programmers. It's preliminary drafts have received

[1] CM-2 is a registered trademark of Thinking Machines Corporation.

excellent reviews from parallel computing experts.

3 List of Publications Resulting from the Grant

Books

- Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. Introduction to Parallel Computing: Algorithm Design and Analysis. *Benjamin/Cummings*, Redwod City, 1994.

Journals

- Ananth Grama, Anshul Gupta and Vipin Kumar. Isoefficiency Function: A Scalability Metric for Parallel Algorithms and Architectures. *IEEE Parallel and Distributed Technology, Special Issue on Parallel and Distributed Systems: From Theory to Practice*, 1(3) August 1993.
- S. Arvindam, Vipin Kumar, V. Nageshwara Rao and Vineet Singh. Automatic test Pattern Generation on Multiprocessors. *Parallel Computing*, 17(12): 1323–1342, December 1991".
- George Karypis and Vipin Kumar. Unstructured Tree Search on SIMD Parallel Computers. *IEEE Transactions on Parallel and Distributed Systems*, 1993 (to appear).
- V. Kumar, S. Shekhar and M. B. Amin. A Scalable Parallel Formulation of Backpropagation Algorithm for Hypercubes and Related Architectures. *IEEE Transactions on Parallel and Distributed Systems*, 1994 (to appear).
- Vipin Kumar, Ananth Grama and V. Nageshwara Rao. Scalable Load Balancing Techniques for Parallel Computers. *Journal of Distributed and Parallel Computing*, 1994 (to appear).
- Vipin Kumar and Anshul Gupta. Analyzing Scalability of Parallel Algorithms and Architectures. *Journal of Parallel and Distributed Computing*, 1993 (to appear).
- Vipin Kumar and Vineet Singh. Scalability of Parallel Algorithms for the All-Pairs Shortest Path Problem. *Journal of Parallel and Distributed Computing (special issue on massively parallel computation)*, 13(2): 124–138, October 1991.
- V. Nageshwara Rao and Vipin Kumar. On the Efficicency of Parallel Backtracking. *IEEE Transactions on Parallel and Distributed Systems*, 4(4): 427–437, April 1993.
- Anshul Gupta and Vipin Kumar. Performance properties of large scale parallel systems. *Journal of Parallel and Distributed Computing (special issue on Supercomputer Performance)*, November 1993.
- Anshul Gupta and Vipin Kumar. The scalability of FFT on Parallel Computers. *IEEE Transactions on Parallel and Distributed Systems*, July 1993 .
- Vineet Singh, Vipin Kumar, Gul Agha and Chris Tomlinson. Scalability of parallel sorting on mesh multicomputers. *International Journal of Parallel Programming*, 20(2), 1991.

Conference Proceedings

- Anshul Gupta, Vipin Kumar and Ahmed Sameh. Performance and Scalability of Preconditioned Conjugate Gradient Methods on Parallel Computers. *Sixth SIAM conference on Parallel Processing for Scientific Computing*, 1992.
- Anshul Gupta and Vipin Kumar. The scalability of Matrix Multiplication Algorithms on Parallel Computers. *Proceedings of International Conference on Parallel Processing*, 1993.
- Ananth Grama and Vipin Kumar. Scalability Analysis of Partitioning Strategies for Finite Element Graphs. *Proceedings of Supercomputing 92, Minneapolis*, 1992.
- Ananth Grama, Vipin Kumar and V. Nageshwara Rao. Experimental Evaluation of Load Balancing Techniques for the Hypercube. *Proceedings of the Parallel Computing 91 Conference*, 1991.
- George Karypis and Vipin Kumar. Efficient Parallel Mappings of a Dynamic Programming Algorithm. Proceedings of the International Parallel Processing Symposium, 1992.
- S. Arvindam, Vipin Kumar and V. Nageshwara Rao. Efficient Parallel Algorithms for Search Problems: Applications in VLSI CAD. *Proceedings of the Frontiers 90 Conference on Massively Parallel Computation*, October 1990.
- D. J. Challou, M. Gini and V. Kumar. Parallel Search Algorithms for Robot Motion Planning. *1993 International Conference on Robotics and Automation*, 1993.
- Vipin Kumar and Anshul Gupta. Analyzing the Scalability of Parallel Algorithms and Architectures: A Survey. *Proceedings of the 1991 International Conference on Supercomputing*, June 1991.
- V. Nageshwara Rao and Vipin Kumar. On the Efficiency of Parallel Ordered Depth-First Search. *Proceedings of the 1991 Conference on Distributed Memory and Concurrent Computers*, May 1991.

Book Chapters

- Ananth Grama V. Kumar and P. Pardalos. Parallel Processing of Discrete Optimization Problems. *Encyclopaedia of Microcomputers*, Marcel Dekker Inc., New York, NY, 1993.

Unpublished Technical Reports

- Ananth Grama and Vipin Kumar. A Survey of Parallel Search Algorithms for Discrete Optimization Problems. *TR-93-11, Department of Computer Science, University of Minnesota, Minneapolis*, 1993.
- Anshul Gupta, Vipin Kumar and Ahmed Sameh. Performance and Scalability of Preconditioned Conjugate Gradient Methods on Parallel Computers. *TR 92-64 University of Minnesota*, 1992.

- Anshul Gupta and Vipin Kumar. The scalability of Matrix Multiplication Algorithms on Parallel Computers. *TR 91-54 University of Minnesota*, 1993.

4 List of Participating Personnel

Anshul Gupta
Ananth Grama
George Karypis

(All of the above are currently working on their PhD under the guidance of Prof. V. Kumar at the University of Minnesota)

References

[1] S. Arvindam, Vipin Kumar, and V. Nageshwara Rao. Floorplan optimization on multiprocessors. In *Proceedings of the 1989 International Conference on Computer Design (ICCD-89)*, 1989. Also published as MCC Technical Report ACT-OODS-241-89.

[2] S. Arvindam, Vipin Kumar, and V. Nageshwara Rao. Efficient parallel algorithms for search problems: Applications in vlsi cad. In *Proceedings of the Frontiers 90 Conference on Massively Parallel Computation*, October 1990.

[3] S. Arvindam, Vipin Kumar, V. Nageshwara Rao, and Vineet Singh. Automatic test pattern generation on multiprocessors. *Parallel Computing*, 17, number 12:1323–1342, December 1991.

[4] Horace P. Flatt. Further applications of the overhead model for parallel systems. Technical Report G320-3540, IBM Corporation, Palo Alto Scientific Center, Palo Alto, CA, 1990.

[5] Horace P. Flatt and Ken Kennedy. Performance of parallel processors. *Parallel Computing*, 12:1–20, 1989.

[6] Ananth Grama, Anshul Gupta, and Vipin Kumar. Isoefficiency function: A scalability metric for parallel algor ithms and architectures. *IEEE Parallel and Distributed Technology, Special Issue on Parallel and Distributed Systems: From Theory to Practice*, 1993 (To be Published). Also available as Technical Report TR-93-24, Department of Computer Science, University of Minnesota and from anonymous ftp site ftp.cs.umn.edu, file users/kumar/isoeff-tutorial.ps.

[7] Ananth Grama and Vipin Kumar. Scalability analysis of partitioning strategies for finite element graphs. In *Proceedings of Supercomputing 92, Minneapolis*, 1992.

[8] Ananth Grama, Vipin Kumar, and V. Nageshwara Rao. Experimental evaluation of load balancing techniques for the hypercube. In *Proceedings of the Parallel Computing 91 Conference*, 1991.

[9] Anshul Gupta and Vipin Kumar. The scalability of Matrix Multiplication Algorithms on parallel computers. Technical Report TR 91-54, Computer Science Department, University of Minnesota, Minneapolis, MN 55455, 1991. A short version of this paper appears in the Proceedings of International Conference on Parallel Processing, 1993.

[10] Anshul Gupta and Vipin Kumar. Performance properties of large scale parallel systems. *Journal of Parallel and Distributed Computing (special issue on Supercomputer Performance)*, November 1993. Also available as Technical Report 92-32, Department of Computer Science, University of Minnesota, Minneapolis, MN.

[11] Anshul Gupta and Vipin Kumar. The scalability of FFT on parallel computers. *IEEE Transactions on Parallel and Distributed Systems*, July 1993 (to appear). Available as a technical report TR 90-53, Computer Science Department, University of Minnesota.

[12] Anshul Gupta, Vipin Kumar, and Ahmed Sameh. Performance and scalability of preconditioned conjugate gradient methods on parallel computers. Technical Report TR 92-64, University of Minnesota, 1992. A short version appears in the sixth SIAM conference on Parallel Processing for Scientific Computing.

[13] W. D. Hillis and G. L. Steele. Data parallel algorithms. *Communications of the ACM*, 29(12):1170–1183, 1986.

[14] George Karypis and Vipin Kumar. Unstructured Tree Search on SIMD Parallel Computers. *IEEE Transactions on Parallel and Distributed Systems*, 1993 (to appear). available as a technical report TR 92-21, Computer Science Department, Un iversity of Minnesota.

[15] V. Kumar, P S. Gopalkrishnan, and L. Kanal (editors). *Parallel Algorithms for Machine Intelligence and Vision*. Springer-Verlag, New York, NY, 1990.

[16] V. Kumar, S. Shekhar, and M. B. Amin. A scalable parallel formulation of backpropagation algorithm for hypercubes and related architectures. *IEEE Transactions on Parallel and Distributed Systems*, 1994 (to appear). available as a technical report TR 92-54, Computer Science Department, Un iversity of Minnesota.

[17] Vipin Kumar, Ananth Grama, Anshul Gupta, and George Karypis. *Introduction to Parallel Computing: Algorithm Design and Analysis*. Benjamin/Cummings, Redwood City, 1994.

[18] Vipin Kumar, Ananth Grama, and V. Nageshwara Rao. Scalable load balancing techniques for parallel computers. Technical report, Technical Report 91-55, Computer Science Department, University of Minnesota, 1991. To appear in Jornal of Distributed and Parallel Computing, 1994.

[19] Vipin Kumar and Anshul Gupta. Analyzing scalability of parallel algorithms and architectures. *Journal of Parallel and Distributed Computing*, 1993 (to appear). A short version of the paper appears in the Proceedings of the 1991 International Conference on Supercomputing,

Germany, and as an invited paper in the Proceedings of 29th Annual Allerton Conference on Communication, Control and Computing, Urbana,IL, October 1991.

[20] Vipin Kumar and V. N. Rao. Scalable parallel formulations of depth-first search. In Vipin Kumar, P. S. Gopalakrishnan, and Laveen Kanal, editors, *Parallel Algorithms for Machine Intelligence and Vision*. Springer-Verlag, New York, NY, 1990.

[21] Vipin Kumar and V. Nageshwara Rao. Parallel depth-first search, part II: Analysis. *International Journal of Parallel Programming*, 16 (6):501–519, December 1987.

[22] Vipin Kumar and V. Nageshwara Rao. Load balancing on the hypercube architecture. In *Proceedings of the 1989 Conference on Hypercubes, Concurrent Computers and Applications*, pages 603–608, 1989.

[23] Vipin Kumar and Vineet Singh. Scalability of Parallel Algorithms for the All-Pairs Shortest Path Problem. *Journal of Parallel and Distributed Computing (special issue on massively parallel computation)*, 13(2):124–138, October 1991. A short version appears in the Proceedings of the International Conference on Parallel Processing, 1990.

[24] Daniel Nussbaum and Anant Agarwal. Scalability of parallel machines. *Communications of the ACM*, 34(3):57–61, 1991.

[25] V. Nageshwara Rao and V. Kumar. Parallel depth-first search, part I: Implementation. *International Journal of Parallel Programming*, 16 (6):479–499, December 1987.

[26] Vineet Singh, Vipin Kumar, Gul Agha, and Chris Tomlinson. Scalability of parallel sorting on mesh multicomputers. *International Journal of Parallel Programming*, 20 (2), 1991. A short version of this paper appears in the Proceedings of the Fifth Interna... nal Parallel Processing Symposium, 1991.

[27] G. L. Wilcox, M. Poliac, and M. Liebman. Protein tertiary structure prediction using a large backpropagation network. In *IJCNN (IEEE/EEC)*, pages 365–369, 1990.

[28] Patrick H. Worley. The effect of time constr..nts on scaled speedup. *SIAM Journal on Scientific and Statistical Computing*, 11(5):838–858, 1990.